# Resampled Belief Networks for Variational Inference

**Ramki Gummadi***
gsrk84@gmail.com

## 1 Introduction

We consider the problem of learning representations using a combination architecture consisting of generative/recognition networks, where the goal is to simultaneously achieve rich representations along with fast inference/training, similar in spirit to [1, 4, 2, 5, 3]. The use of a directed belief network for the recognition part helps in training the generative model, but is also useful purely as a tool for efficient inference (e.g.[6, 7]). For a generative model density $p$ parametrized by $\boldsymbol{\theta}$, and any other density $q$ (e.g. the recognition density, parametrized by $\boldsymbol{\phi}$), conditioning on the evidence $\boldsymbol{v}$, we have the following equations (reviewed in Section 7.1):

$$\log p(\boldsymbol{v}) = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v})}\left[\log p(\boldsymbol{h},\boldsymbol{v})\right] = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\log \frac{p(\boldsymbol{h},\boldsymbol{v})}{q(\boldsymbol{h}|\boldsymbol{v})}\right] + \mathtt{KL}(q(\boldsymbol{h}|\boldsymbol{v})||p(\boldsymbol{h}|\boldsymbol{v}))$$

$$\nabla_\theta \log p(\boldsymbol{v}) = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v})}\left[\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{h},\boldsymbol{v})\right] = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\nabla_{\boldsymbol{\theta}} \log p(\boldsymbol{h},\boldsymbol{v})\right] + \nabla_{\boldsymbol{\theta}}\mathtt{KL}\left(q(\boldsymbol{h}|\boldsymbol{v})||p(\boldsymbol{h}|\boldsymbol{v})\right)$$

Although the generative and the recognition network each individually contains a representation of the data, the task of jointly training them may be viewed as an approximation to training the generative model alone. Variational methods are able to drop the second term in the above gradient for $\boldsymbol{\theta}$ by instead estimating $\nabla_{\boldsymbol{\phi}}\mathtt{KL}\left(q(\boldsymbol{h}|\boldsymbol{v})||p(\boldsymbol{h}|\boldsymbol{v})\right)$ and then performing gradient ascent on an expanded collection of parameters $(\boldsymbol{\theta}, \boldsymbol{\phi})$. However, the path taken by $\theta$ during gradient ascent on the *evidence lower bound* in variational methods (which corresponds to dropping the KL divergence) clearly depends on the class over which $q$ is optimized. In mean field methods [8] that use independence assumptions, the parameters $\boldsymbol{\phi}$ need to be recomputed from scratch for each inference instance. By contrast, using a directed recognition network [1, 4, 2, 3]) with parametrized edge weights allows learning an inverse factorization of the generative model directly. However: (1) it may not necessarily contain the density implied for inference; and (2) training the recognition model parameters is a challenging problem, even with a fixed $\boldsymbol{\theta}$.

We propose a class of parametrized densities called Resampled Belief Networks (RBN), which are: (1) at least theoretically, capable of representing the exact inverse and (2) introduce an additional lever to control the variance of gradient estimates. An RBN augments the recognition directed belief network with a resampling scheme designed to tighten the gap between the recognition and generative densities. In Section 2, we characterize the density achieved by an RBN in terms of the constituent directed network and the resampling rules. Using this characterization, in Sections 3 and 4 we develop a scheme to break down a given recognition/generative model into corresponding resampling constraints. Another motivation for this approach is to achieve tradeoff points along a continuum between monte carlo approaches with perfect accuracy but impractical computational expense, and variational approximations with much better computational expense.

## 2 Resampled Belief Networks (RBN)

Let $R$ be a directed belief network on $\boldsymbol{x} = (x_1, \ldots, x_n)$ that samples from $x_1$ to $x_n$ using: $r(\boldsymbol{x}) = \prod_{i=1}^{n} r_i(x_i|\boldsymbol{x}_{pa_r(i)})$. By an RBN, we refer to the sampler defined in Algorithm 1,

*This research was primarily done while the author was with the Department of Computing Science, University of Alberta. Current homepage: http://ramki-gummadi.github.io/

by augmenting $R$ with (i) *resampling constraints* specified using a collection of constraint functions, $C = \{C_1, \ldots, C_n\}$, where each constraint[1], $C_i : \boldsymbol{x}_{[i]} \mapsto (0, 1]$; and (ii) *jump rules* $\Delta = \{\Delta_1, \ldots, \Delta_n\}$ where $\Delta_i \in \{0, \ldots i-1\}$ specifies the node from which resampling needs to be done when constraint $i$ fails.

**Algorithm 1** Sampling definition of $\texttt{RBN}(R, C, \Delta)$.

$i = 1$.
**while** $i \leq n$ **do**
  $x_i \leftarrow$ sample from $r_i(.|\boldsymbol{x}_{pa_r(i)})$.
  $u \sim U[0, 1]$.
  **if** $u > C_i(\boldsymbol{x})$ **then**
    $i \leftarrow \Delta_i$.
  $i \leftarrow i + 1$.
Output sample $\boldsymbol{x} = (x_1, \ldots, x_n)$.

**Theorem 1** (Density characterization)**.** *Let $Z_i(\boldsymbol{x}) \triangleq \mathbb{E}[C_i(\boldsymbol{x})|(x_1, \ldots, x_{\Delta_i})]$, where the expectation is with respect to the density induced on $(x_1, \ldots, x_i)$ prior to constraint $i$. Then, the sampling density of Algorithm 1 satisfies $q(\boldsymbol{x}) = r(\boldsymbol{x}) \frac{\prod_i C_i(\boldsymbol{x})}{\prod_i Z_i(\boldsymbol{x})}$.*

Note that $Z_i$ denotes the probability that a randomly generated sample is accepted by constraint $i$ as a function of the sampled nodes prior to the resampling point.

**Corollary 2.** *If $Z_i(x_1, \ldots, x_{\Delta_i}) = Z_i$, i.e. independent of its arguments, then $q(\boldsymbol{x}) = \frac{1}{Z} r(\boldsymbol{x}) \prod_i C_i(\boldsymbol{x})$, where $Z$ is the appropriate normalizing constant. A useful special case is that of always resampling from the start, i.e. $\Delta_i = 0 \ \forall\, i$.*

## 3 Generative/Recognition Network architecture with RBN

Assume all variables are binary and let $p(\boldsymbol{x})$ denote the distribution implied by a generative model, parametrized by the vector $\boldsymbol{\theta}$. Let the recognition model contain a directed belief network with distribution $r(\boldsymbol{x})$, whose parameters are contained in the vector $\boldsymbol{\phi}$. The energy functions for relevant distributions will be composed from $a_i(\boldsymbol{x})$ and $b_i(\boldsymbol{x})$ defined below, where $pa(i)$ refers to the generative model.

$$a_i(\boldsymbol{x}) = \begin{cases} \sum_{j \in pa(i)} \theta_{ji} x_j \text{ if } x_i = 1 \\ -\sum_{j \in pa(i)} \theta_{ji} x_j \text{ if } x_i = 0 \end{cases}, \quad b_i(\boldsymbol{x}) = \begin{cases} \sum_{j<i} \phi_{ji} x_j \text{ if } x_i = 1 \\ -\sum_{j<i} \phi_{ji} x_j \text{ if } x_i = 0 \end{cases} \tag{1}$$

Using the shorthand operator $f^+(\boldsymbol{x}) \triangleq log(1 + e^{f(\boldsymbol{x})})$, we will rewrite the usual expression for the distributions as follows.

$$r(\boldsymbol{x}) = \prod_{i \geq 0} \frac{e^{-(\sum_{j<i} \phi_{ji} x_j) x_i}}{1 + e^{-(\sum_{j<i} \phi_{ji} x_j)}} = \prod_{i \geq 0} e^{-b_i^+(\boldsymbol{x})}, \text{ and } p(\boldsymbol{x}) = \frac{1}{Z_p} \prod_i e^{-a_i^+(\boldsymbol{x})} \tag{2}$$

The identity is easily verified by considering $x_i \in \{0, 1\}$ separately.[2] Let the constraint functions be defined by their negative log likelihoods, $c_j(\boldsymbol{x}) \geq 0$ over the constraint index $j$, i.e., $C_j(\boldsymbol{x}) = e^{-c_j(\boldsymbol{x})}$. For now, we will only consider jump rules with $\Delta_i = 0 \ \forall i$, although this is not necessary to satisfy Equation (3) as long as Corollary 2 is valid. From Theorem 1, the recognition network samples from:

$$q(\boldsymbol{x}) = \frac{1}{Z_q} \prod_{i \geq 0} e^{-b_i^+(\boldsymbol{x})} \prod_j e^{-c_j(\boldsymbol{x})} \triangleq \frac{\gamma_q(\boldsymbol{x})}{Z_q} \tag{3}$$

[1] Let $[i] \triangleq \{1, 2, \ldots, i\}$. Furthermore, we abuse notation and write the element, rather than the set which is being mapped, for clarity. Note that $C_i$ are functions that depend only on the first $i$ variables of $\boldsymbol{x}$. However, for reducing notation clutter, we use $C_i(\boldsymbol{x})$ with the implicit recognition that the components of $x$ outside $[i]$ are irrelevant.

[2] Note that $Z_p = 1$, but we write it in the above equation explicitly to reuse the equation with separate hidden and visible variables. For that purpose, the index $i$ may be allowed to have terms beyond the indices of the sampling variables.

Note that rejection sampling is analogous to using a single constraint function, $c(\boldsymbol{x}) = c + \sum_i a_i^+(\boldsymbol{x}) - \sum_{i\geq 0} b_i^+(\boldsymbol{x})$, where $c$ is any constant chosen large enough to ensure that $c(\boldsymbol{x}) \geq 0 \ \forall \boldsymbol{x}$. An interpretation of the subclass of jump rules for which $\Delta_i = 0$, is to view them as approximately breaking down a large rejection event into smaller sequential pre-emptive steps, thereby improving the sampling efficiency, possibly exponentially.

**Proposition 3.**

$$\textit{Let:} \quad \mathcal{A}(\boldsymbol{x}) \triangleq \sum_i a_i^+(\boldsymbol{x}) - \sum_{i\geq 0} b_i^+(\boldsymbol{x}) - \sum_j c_j(\boldsymbol{x}), \textit{ and } \bar{\mathcal{A}}(\boldsymbol{x}) \triangleq \mathcal{A}(\boldsymbol{x}) - \mathbb{E}_q\left[\mathcal{A}(\boldsymbol{x})\right] \tag{4}$$

$$\textit{Then,} \quad \mathtt{KL}(q||p) = \log\left(\mathbb{E}_q\left[e^{-\bar{\mathcal{A}}(\boldsymbol{x})}\right]\right), \textit{ and } \nabla_{\boldsymbol{\phi}}\,\mathtt{KL}(q||p) = \mathtt{COV}\left(\mathcal{A}(\boldsymbol{x}), \nabla_{\boldsymbol{\phi}}\mathcal{A}(\boldsymbol{x})\right) \tag{5}$$

*Furthermore, when* $\mathtt{KL}(q||p) \approx 0$*, an approximation to the KL divergence is* $\mathtt{KL}(q||p) \approx \mathtt{VAR}(\mathcal{A}(\boldsymbol{x}))/2$.

The proof is provided via Theorem 7 in the Appendix. In fact, it is worth noting that Theorem 7 is stated for arbitrary unnormalized distributions and not specifically for RBNs. The claim about the approximation is justified by making use of the second order Taylor approximation $e^{-x} \approx 1 - x + x^2/2$ in Equation (5). While Equation (5) suggests the importance of controlling the variance of $\mathcal{A}(\boldsymbol{x})$ to estimate gradients for the recognition network, Proposition 3 also suggests that achieving a low variance for $\mathcal{A}(\boldsymbol{x})$ would require having access to a sampler with an approximately accurate inferential density, implying a chicken and egg situation. This justifies the claim made in the introduction that RBNs contain a lever for variance control of the gradient estimates.

## 4 Derivation of Constraints from a Generative Model

In this Section, we use the above framework to propose one possible general design for the resampling constraints without making any assumptions on the generative model. The basic observation underlying this proposal is that, under the target sampling density $p(\boldsymbol{x})$, the functions $a_i^+(\boldsymbol{x})$ have a strong, unconditional concentration property. This property, formalized in Lemma 4, therefore suggests that whenever a generated sample violates this concentration property by an appropriately large margin, it could be considered a credible signal to resample.

**Lemma 4.** *$a_i^+(\boldsymbol{x}) < 0.7 + 0.75k$ with probability at least $1 - 1/k^2$, uniformly across $\boldsymbol{\theta}, i$ and sigmoid generative sample densities $p(\boldsymbol{x})$ defined in Equation 2 (with $Z_p = 1$).*

The proof is provided in the appendix. Let: $S_t = \{i : pa(i) \in [t] \text{ and } pa(i) \notin [t-1]\}$, for $t \in [n]$. That is, $S_t$ represents the constraints that are revealed at step $t$ in the recognition network. It is helpful to rewrite Equation (4) as:

$$\mathcal{A}(\boldsymbol{x}) = \sum_{t=1}^{n} A_t(\boldsymbol{x}) - \sum_j c_j(\boldsymbol{x}), \text{ where } A_t(\boldsymbol{x}) \triangleq \sum_{i\in S_t} a_i^+(\boldsymbol{x}) - b_t^+(\boldsymbol{x}) \tag{6}$$

For an arbitrarily general factorization of $p(\boldsymbol{x})$ with no structure, $S_t$ may be non-empty only when $t = n$, but it could be subject to a more uniform "revelation schedule" in structured factorizations, e.g. layered networks. The concentration property for individual $a_i^+(\boldsymbol{x})$ in Lemma 4 can be generalized to the functions $A_t(\boldsymbol{x})$, even though the different $a_i(\boldsymbol{x})$ are correlated, to obtain appropriate bounds that only depend on the size of $S_t$. This motivates the class of constraint functions in Equation (7), where $\{T_t : t \in [n]\}$ may be considered as additional parameters complementary to the edge weights $\phi_{lm}$ in determining the recognition density $q$.

$$c_t(\boldsymbol{x}) = \left(A_t(\boldsymbol{x}) - T_t\right)^+ \quad \forall t \in [n] \tag{7}$$

This implies: $\mathcal{A}(\boldsymbol{x}) = \sum_{t=1}^{n}\left(A_t(\boldsymbol{x}) - \left(A_t(\boldsymbol{x}) - T_t\right)^+\right)$. From this equation, it is clear that setting $T_t$ small enough will ensure that $\mathtt{Var}(\mathcal{A}(\boldsymbol{x})) \approx 0$, which in turn implies that $\mathtt{KL}(q||p) \approx 0$. However, this could be infeasible due to an untenable sampling efficiency. By contrast, setting $T_t$ large disables all constraints, and the sampling density of the RBN remains unaltered from $R$.

## 5 Experimental Observations

We run numerical experiments using two synthetic binary 100 node sigmoid belief network generative/recognition architectures, $(R_i, G_i)$ for $i \in \{\texttt{rev}, \texttt{nr}\}$, where $R_i$ is part of the recognition density and $G_i$ is its generative counterpart. Our initial focus is not directly on training $(R_i, G_i)$, but to first study the effects of introducing the proposed resampling framework on top of the standard architecture. $G_{\texttt{rev}}$ is chosen to correspond to a reversible markov chain, due to its property that its exact inverse factorization has identical parameters as $G_i$. However, the exclusively one step dependencies might be expected to result in weak constraints across nodes, suggesting that this may be a pathological case for demonstrating any potential benefits of resampling. $R_{\texttt{rev}}$ was fixed to be a small perturbation from the perfect inverse factorization. By contrast, $G_{\texttt{nr}}$ had randomly sampled edge weights from each node upto its 10 immediate parents. $R_{\texttt{nr}}$ had full connectivity with randomly sampled edge weights. The choice of the thresholds $T_t$, left unspecified in Section 4, is chosen according to the following heuristic for both experiments: First, for each $t \in [n]$, we form Monte Carlo estimates for the mean, $\hat{\mu}_t$ and standard deviation, $\hat{\sigma}_t$, for samples of $A_t(\boldsymbol{x})$ defined in Equation (6) under $G_i$. Given a uniform (across $t$) control parameter, called gap, we then choose $T_t = \hat{\mu}_t + \texttt{gap} * \hat{\sigma}_t$. Finally, gap is adjusted according to a multiplicative rate control heuristic that responds to the sampling efficiency. We find that the control variable gap is more instrumental in the case of nrev. The following three statistics are shown in the appendix in Figure 1 across the trajectory of the gap variable for the resulting RBNs, on each of rev and nrev: (1) NLL (2) Sample efficiency (3) $\texttt{VAR}(\mathcal{A}(\boldsymbol{x}))/2$ (which may be considered a proxy for the KL distance achieved as explained in Proposition 3).

## 6 Amortization of Inference and Training Equations

In this section, we consider the problem of training the RBN. Split $\boldsymbol{x} = (\boldsymbol{h}, \boldsymbol{v})$, where $\boldsymbol{v} = (x_1, \ldots, x_i)$ and $\boldsymbol{h} = (x_{i+1}, \ldots, x_n)$ represent the visible/hidden parts, and restrict $\boldsymbol{\phi}$ to those parameters that only affect the conditional density of $\boldsymbol{h}$ given $\boldsymbol{v}$. The gradient in Equation (5) can be rewritten using the law of total covariance as:

$$\nabla_{\boldsymbol{\phi}} \, \texttt{KL}(q||p) = \mathbb{E}_{\boldsymbol{v}} \left[ \texttt{COV}_{\boldsymbol{h}}(\mathcal{A}(\boldsymbol{x}), \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})) \right] + \texttt{COV}_{\boldsymbol{v}} \left( \mathbb{E}_{\boldsymbol{h}}[\mathcal{A}(\boldsymbol{x})], \mathbb{E}_{\boldsymbol{h}}[\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})] \right) \tag{8}$$

In the above equation, subscripts of operators refer to the variables over which we are not conditioning, i.e. considering as random variables. From the proof of Lemma 7, we can write $\mathbb{E}_{\boldsymbol{h}} \left[ \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{v}, \boldsymbol{h}) \right] = \nabla_{\boldsymbol{\phi}} \log Z_{q_{\boldsymbol{v}}}$, where $Z_{q_{\boldsymbol{v}}} = \sum_{\boldsymbol{h}} \gamma_q(\boldsymbol{v}, \boldsymbol{h})$. By defining $q_{\boldsymbol{v}}, p_{\boldsymbol{v}}$ as the conditional densities of $q$ and $p$ on $\boldsymbol{h}$ given $\boldsymbol{v}$, and applying $\nabla_{\boldsymbol{\phi}} \, \texttt{KL}(q_{\boldsymbol{v}}||p_{\boldsymbol{v}}) = \texttt{COV}_{\boldsymbol{h}}(\mathcal{A}(\boldsymbol{v}, \boldsymbol{h}), \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{v}, \boldsymbol{h}))$, we may write Equation (8) as:

$$\nabla_{\boldsymbol{\phi}} \, \texttt{KL}(q||p) = \mathbb{E}_{\boldsymbol{v}} \left[ \nabla_{\boldsymbol{\phi}} \, \texttt{KL}(q_{\boldsymbol{v}}||p_{\boldsymbol{v}}) \right] + \texttt{COV}_{\boldsymbol{v}} \left( \mathbb{E}_{\boldsymbol{h}}[\mathcal{A}(\boldsymbol{x})], \nabla_{\boldsymbol{\phi}} \log Z_{q_{\boldsymbol{v}}} \right) \tag{9}$$

From the above equation, we see that $Z_{q_{\boldsymbol{v}}}$ being independent of $\boldsymbol{v}$ (can still depend on $i$ or the edge weights) is a sufficient condition to equate the second term is zero. In this case, the parameter training can be averaged out over different inference instances. For an RBN, $Z_{q_{\boldsymbol{v}}}$ is equal to the probability of acceptance of a sample $\boldsymbol{x}$ starting from $\boldsymbol{v}$, provided that $\Delta_j = i \;\forall\; j > i$. Therefore, any node in the network where the probability of acceptance for samples generated from that point is independent of the values of the nodes that were sampled previously can serve as an amortization point for training the parameters of the recognition network. Since the acceptance probabilities in an RBN can be controlled via threshold parameters, this suggests one possible way to exploit this observation in practice. Theorem 5 provides a summary of the derivations for the gradients of the training parameters in an RBN.

**Theorem 5.** *Let $\phi_{lm}$ be the weight parameter connecting node $x_l$ to node $x_m$ ($l < m$) in the recognition network, R. Let $T_m$ denote the threshold parameter in $c_m(\boldsymbol{x})$ according to Equation 7 for the* RBN. *With $\sigma(\boldsymbol{x}) \triangleq (1 + e^{-\boldsymbol{x}})^{-1}$, we get the following expressions for gradients, where we define $\mu_m(\boldsymbol{x})$ as the conditional expected value of $x_m$ for the belief network R, given all its parents.*

$$-\frac{\partial \texttt{KL}(q||p)}{\partial \phi_{lm}} = \texttt{COV} \left( \mathcal{A}(\boldsymbol{x}), (1 - \mu_m(\boldsymbol{x})) x_l x_m \sigma \left( T_m - A_m(\boldsymbol{x}) \right) \right)$$

$$\frac{\partial \texttt{KL}(q||p)}{\partial T_m} = \texttt{COV} \left( \mathcal{A}(\boldsymbol{x}), \sigma \left( A_m(\boldsymbol{x}) - T_m \right) \right)$$

An interesting direction for future work is to evaluate simultaneous training of edge weight parameters alongside learning the resampling thresholds and comparing it with the heuristic learning rule that was used in Section 5.

# 7 Appendix

This Section provides proof details, as well as a figure reporting some numerical observations in experiments.

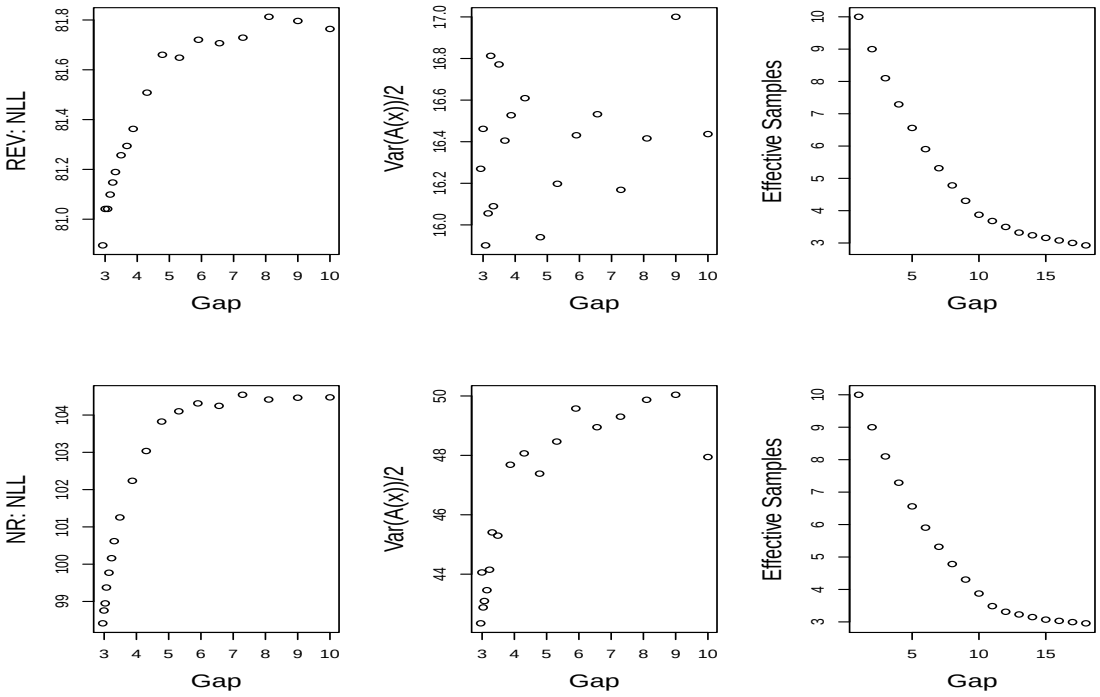


Figure 1: The top row figures show the statistics for a reversible Markov Chain on 100 binary nodes, for which the conditionals $p(x_{i+1}|\boldsymbol{x}_{1:i}) = p(x_{i+1}|x_i)$, correspond to the transition matrix $\begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$. The prior for the first node was chosen to achieve reversibility, i.e. the stationary distribution. The bottom row depicts results from an experiment where the generative network was fixed using a distribution where each node has a conditional that depends on the previous ten sampled nodes, where the corresponding edge weights were chosen by sampling from a standard normal. The gap variable refers to a uniform control parameter for the threshold parameters across all nodes according to the mechanism described in Section 5. Each depicted point is an average over $10,000$ samples of the corresponding RBN resulting from the threshold rule defined by the gap variable. NLL refers to the negative log likelihood of samples generated by the RBN evaluated with respect to the corresponding generative density. The effective samples depict the factor overhead that results from resampling. All resampling rules implemented correspond to $\Delta_i = 0 \ \forall i$.

**Theorem** (Density characterization)**.** *Let $Z_i(\boldsymbol{x}) \triangleq \mathbb{E}[C_i(\boldsymbol{x})|(x_1, \ldots, x_{\Delta_i})]$, where the expectation is with respect to the density induced on $(x_1, \ldots, x_i)$ prior to constraint $i$. Note that $Z_i$ denotes the probability that constraint $i$ fails, as a function of the nodes prior to the resampling point. Then, the sampling density of Algorithm 1 satisfies $q(\boldsymbol{x}) = r(\boldsymbol{x}) \frac{\prod_i C_i(\boldsymbol{x})}{\prod_i Z_i(\boldsymbol{x})}$.*

**Proof.** [Proof of Theorem 1] Wherever convenient, we will use $\boldsymbol{x}_{1:i}$ to denote $(x_1, \ldots, x_i)$. Let $q_k(x_{1:k})$ be the sampling density of $\boldsymbol{x}_{1:k}$, at the point where $i = k$, just prior to Step 7. Therefore,

the density we are evaluating, $q(\boldsymbol{x})$, is equal to $q_n(\boldsymbol{x})$. Note that $q_k(\boldsymbol{x}_{1:k})$ is not necessarily equal to the marginal density of $q_n$ on $\boldsymbol{x}_{1:k}$. Let $q_k(\boldsymbol{x}_{1:i})$ denote the marginal density of $q_k$ on $\boldsymbol{x}_{1:i}$, and let $q_k(\boldsymbol{x}_{i+1:k}|\boldsymbol{x}_{1:i})$ denote the appropriate conditional density. The proof will use induction on $n$. The base case is trivial, and assume that the induction hypothesis holds for $n-1$. Let $\texttt{accept} \triangleq \mathbb{1}\{u \leq C_n(\boldsymbol{x})\}$ in Step 5, so that $P(\texttt{accept}|X) = \mathbb{E}[C_n(\boldsymbol{x})|X]$ for any random variable $X$. Conditioning on $\boldsymbol{x}_{1:\Delta_n}$:

$$\begin{aligned}
q_n((x_{\Delta_n+1},\ldots,x_n)|\boldsymbol{x}_{1:\Delta_n}) &= P\left((x_{\Delta_n+1},\ldots,x_n)|\texttt{accept},\boldsymbol{x}_{1:\Delta_n}\right) \\
&= P\left(((x_{\Delta_n+1},\ldots,x_n),\texttt{accept})|\boldsymbol{x}_{1:\Delta_n}\right)/P\left(\texttt{accept}|\boldsymbol{x}_{1:\Delta_n}\right) \\
&= q_{n-1}((x_{\Delta_n+1},\ldots,x_{n-1})|\boldsymbol{x}_{1:\Delta_n})r(x_n|\boldsymbol{x}_{1:n-1})C_n(\boldsymbol{x})/Z_n(\boldsymbol{x}_{1:\Delta_n})
\end{aligned}$$

Noting that the marginals of $q_n$ and $q_{n-1}$ are identical on $\boldsymbol{x}_{1:\Delta_n}$,

$$\begin{aligned}
q_n(x_{1:n}) &= q_n(\boldsymbol{x}_{1:\Delta_n})q_n((x_{\Delta_n+1},\ldots,x_n)|\boldsymbol{x}_{1:\Delta_n}) \\
&= q_{n-1}(\boldsymbol{x}_{1:\Delta_n})q_{n-1}((x_{\Delta_n+1},\ldots,x_{n-1})|\boldsymbol{x}_{1:\Delta_n})r(x_n|\boldsymbol{x}_{1:n-1})C_n(\boldsymbol{x})/Z_n(\boldsymbol{x}_{1:\Delta_n}) \\
&= q_{n-1}(\boldsymbol{x}_{1:n-1})r(x_n|\boldsymbol{x}_{1:n-1})C_n(\boldsymbol{x})/Z_n(\boldsymbol{x}_{1:\Delta_n})
\end{aligned}$$

The proof follows by appealing to the induction hypothesis in the last equation. □

**Remark 6.** *The constraints were defined over the index set $[n]$ for clarity of presentation. However, the characterization identified also generalizes to an arbitrary index set, with $C_j : \boldsymbol{x}_{A_j} \mapsto (0,1]$. In this case, Step 5 is a loop over $\{j : A_j \subseteq [i] \text{ and } A_j \not\subseteq [i-1]\}$. A natural resampling rule to consider here would be: $\Delta_j = \min\{A_j\} - 1$, although the characterization remains valid regardless.*

**Theorem 7.** *Suppose $p(\boldsymbol{x}) = \gamma_p(\boldsymbol{x})/Z_p$ and $q(\boldsymbol{x}) = \gamma_q(\boldsymbol{x})/Z_q$, where only $q$ depends on $\boldsymbol{\phi}$. Let $\mathcal{A}(\boldsymbol{x}) \triangleq \log\gamma_q(\boldsymbol{x}) - \log\gamma_p(\boldsymbol{x})$; and $\bar{\mathcal{A}}(\boldsymbol{x}) \triangleq \mathcal{A}(\boldsymbol{x}) - \mathbb{E}_q\left[\mathcal{A}(\boldsymbol{x})\right]$. Then:*

$$\texttt{KL}(q||p) = \log\left(\mathbb{E}_q\left[e^{-\bar{\mathcal{A}}(\boldsymbol{x})}\right]\right) \tag{10}$$

$$\nabla_{\boldsymbol{\phi}}\,\texttt{KL}(q||p) = \texttt{COV}_q\left(\mathcal{A}(\boldsymbol{x}), \nabla_{\boldsymbol{\phi}}\mathcal{A}(\boldsymbol{x})\right) \tag{11}$$

*where the subscript $q$ refers to the density used to generate the samples of $\boldsymbol{x}$.*

**Proof.**

$$\begin{aligned}
\texttt{KL}(q||p) &= \log Z_p - \log Z_q + \mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x}) - \log\gamma_p(\boldsymbol{x})\right] \\
&= \log\left(\sum_{\boldsymbol{x}}\frac{\gamma_p(\boldsymbol{x})}{Z_q}\right) + \mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x}) - \log\gamma_p(\boldsymbol{x})\right] \\
&= \log\left(\mathbb{E}_q\left[\frac{\gamma_p(\boldsymbol{x})}{\gamma_q(\boldsymbol{x})}\right]\right) + \mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x}) - \log\gamma_p(\boldsymbol{x})\right] \\
&= \log\left(\mathbb{E}_q\left[e^{-\mathcal{A}(\boldsymbol{x})}\right]\right) + \mathbb{E}_q\left[\mathcal{A}(\boldsymbol{x})\right] = \log\left(\mathbb{E}_q\left[e^{-\bar{\mathcal{A}}(\boldsymbol{x})}\right]\right)
\end{aligned}$$

For the gradients, differentiating the first equality for $\texttt{KL}(q||p)$, we get: $\nabla_{\boldsymbol{\phi}}\,\texttt{KL}(q||p) = D_1 - D_2 - D_3$, where $D_1 = \nabla_{\boldsymbol{\phi}}\mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x})\right]$, $D_2 = \nabla_{\boldsymbol{\phi}}\mathbb{E}_q\left[\log\gamma_p(\boldsymbol{x})\right]$, $D_3 = \nabla_{\boldsymbol{\phi}}\log Z_q$.

$$\begin{aligned}
D_1 = \nabla_{\boldsymbol{\phi}}\mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x})\right] &= \sum_{\boldsymbol{x}}\nabla_{\boldsymbol{\phi}}\left[q(\boldsymbol{x})\log\gamma_q(\boldsymbol{x})\right] \\
&= \sum_{\boldsymbol{x}}\left(\frac{q(\boldsymbol{x})}{\gamma_q(\boldsymbol{x})}\nabla_{\boldsymbol{\phi}}\gamma_q(\boldsymbol{x}) + \log\gamma_q(\boldsymbol{x})\,\nabla_{\boldsymbol{\phi}}q(\boldsymbol{x})\right) \\
&= \frac{1}{Z_q}\nabla_{\boldsymbol{\phi}}Z_q + \sum_{\boldsymbol{x}}q(\boldsymbol{x})\log\gamma_q(\boldsymbol{x})\nabla_{\boldsymbol{\phi}}\log q(\boldsymbol{x}) \\
&= D_3 + \mathbb{E}_q\left[\log\gamma_q(\boldsymbol{x})\nabla_{\boldsymbol{\phi}}\log q(\boldsymbol{x})\right]
\end{aligned}$$

Similarly, $D_2 = \mathbb{E}_q\left[\log\gamma_p(\boldsymbol{x})\nabla_{\boldsymbol{\phi}}\log q(\boldsymbol{x})\right]$, which implies:

$$\nabla_{\boldsymbol{\phi}}\,\texttt{KL}(q||p) = \mathbb{E}_q\left[\left(\log\gamma_q(\boldsymbol{x}) - \log\gamma_p(\boldsymbol{x})\right)\nabla_{\boldsymbol{\phi}}\log q(\boldsymbol{x})\right]$$

Note that $\nabla_{\boldsymbol{\phi}} \log q(\boldsymbol{x}) = \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x}) - D_3 = \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x}) - \mathbb{E}_q \left[\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})\right]$, where:

$$D_3 = \nabla_{\boldsymbol{\phi}} \log Z_q = \frac{1}{Z_q} \sum_{\boldsymbol{x}} \nabla_{\boldsymbol{\phi}} \gamma_q(\boldsymbol{x}) = \frac{1}{Z_q} \sum_{\boldsymbol{x}} \gamma_q(\boldsymbol{x}) \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x}) = \mathbb{E}_q \left[\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})\right]$$

This implies:

$$\nabla_{\boldsymbol{\phi}} \, \mathtt{KL}(q||p) = \mathbb{E}_q \left[ \left(\log \gamma_q(\boldsymbol{x}) - \log \gamma_p(\boldsymbol{x})\right) \overline{\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})} \right] \tag{12}$$

$$= \mathtt{COV} \left(\log \gamma_q(\boldsymbol{x}) - \log \gamma_p(\boldsymbol{x}), \nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x})\right) \tag{13}$$

which completes the proof since $\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x}) = \nabla_{\boldsymbol{\phi}} \mathcal{A}(\boldsymbol{x})$. □

**Lemma.** *$a_i^+(\boldsymbol{x}) < 0.7 + 0.75k$ with probability at least $1 - 1/k^2$, uniformly across $\boldsymbol{\theta}$, $i$ and sigmoid generative sample densities $p(\boldsymbol{x})$ defined in Equation 2 (with $Z_p = 1$).*

**Proof.** [Proof of Lemma 4] Denote $\alpha_i \triangleq \left(\sum_{j \in pa(i)} \theta_{ji} x_j\right)$. Then, $\mathtt{VAR}(a_i^+(\boldsymbol{x})) \leq \mathbb{E}[(a_i^+(\boldsymbol{x}))^2] = \mathbb{E}[\mathbb{E}[(a_i^+(\boldsymbol{x}))^2 | \alpha_i]] < 0.6$ from Lemma 8. Similarly, $\mathbb{E}[a_i^+(\boldsymbol{x})|\alpha_i] \leq \ln 2$, again from Lemma 8. The claim then follows as an application of Chebychev's inequality. It is also feasible to obtain tighter concentration bounds using exponential versions of the inequality. □

**Lemma 8.** *Suppose $X = \left[\alpha \left(\mathbb{1}\{x = 1\} - \mathbb{1}\{x = 0\}\right)\right]^+$, for any constant $\alpha$, where $x$ has the sigmoid distribution with probability $\frac{e^{-\alpha x}}{1+e^{-\alpha}}$, for $x \in \{0, 1\}$ and $a^+ \triangleq \log(1 + e^a)$. Then: $0 \leq \mathbb{E}[X] < \log 2 \approx 0.693$ and $\mathbb{E}[X^2] < 0.6$ and $\mathtt{VAR}(X) < 0.5$ for any $\alpha$.*

**Proof.** The proof of this claim can be obtained by explicitly computing the below functions of $\alpha$ for $k = 1, 2$:

$$\mathbb{E}[X^k] = \frac{\log^k(1 + e^{\alpha})}{1 + e^{\alpha}} + \frac{\log^k(1 + e^{-\alpha})}{1 + e^{-\alpha}}$$

Note also, that arbitrarily high moments of $X$ have bounds that are independent of $\alpha$. □

**Theorem.** *Let $\phi_{lm}$ be the weight parameter connecting node $x_l$ to node $x_m$ ($l < m$) in the recognition network, $R$. Let $T_m$ denote the threshold parameter in $c_m(\boldsymbol{x})$ according to Equation 7 for the* RBN. *With $\sigma(\boldsymbol{x}) \triangleq (1 + e^{-\boldsymbol{x}})^{-1}$, we get the following expressions for gradients, where we define $\mu_m(\boldsymbol{x})$ as the conditional expected value of $x_m$ for the belief network $R$, given all its parents.*

$$-\frac{\partial \mathtt{KL}(q||p)}{\partial \phi_{lm}} = \mathtt{COV} \left(\mathcal{A}(\boldsymbol{x}), (1 - \mu_m(\boldsymbol{x})) x_l x_m \sigma \left(T_m - A_m(\boldsymbol{x})\right)\right)$$

$$\frac{\partial \mathtt{KL}(q||p)}{\partial T_m} = \mathtt{COV} \left(\mathcal{A}(\boldsymbol{x}), \sigma \left(A_m(\boldsymbol{x}) - T_m\right)\right)$$

**Proof.** [Proof of Theorem 5]

$$-\nabla_{\boldsymbol{\phi}} \log \gamma_q(\boldsymbol{x}) = \nabla_{\boldsymbol{\phi}} \left( \sum_{i \geq 0} b_i^+(\boldsymbol{x}) + \sum_j c_j(\boldsymbol{x}) \right)$$

$$= \sum_{i \geq 0} \sigma(b_i(\boldsymbol{x})) \nabla_{\boldsymbol{\phi}} b_i(\boldsymbol{x}) + \sum_j \nabla_{\boldsymbol{\phi}} c_j(\boldsymbol{x}), \text{ where } \sigma(\boldsymbol{x}) = \left(1 + e^{-\boldsymbol{x}}\right)^{-1}$$

Consider the parameter $\phi_{lm}$, which is the weight parameter connecting node $x_l$ to node $x_m$ in the recognition network, where $l < m$. The only non-zero term in the first summation is for $i = m$. Recalling Equation (1), we obtain:

$$-\frac{\partial \log \gamma_q(\boldsymbol{x})}{\partial \phi_{lm}} = \sigma(b_m(\boldsymbol{x})) x_l x_m + \sum_j \frac{\partial c_j(\boldsymbol{x})}{\partial \phi_{lm}} = (1 - \mu_m(\boldsymbol{x})) x_l x_m + \sum_j \frac{\partial c_j(\boldsymbol{x})}{\partial \phi_{lm}} \tag{14}$$

where, we define $\mu_m(\boldsymbol{x})$ as the conditional expected value of $x_m$ given all its parents in the recognition network, $R$, so that $\sigma(b_m(\boldsymbol{x})) = 1 - \mu_m(\boldsymbol{x})$.

Referring back to Equation (14), we see from Equation (7) that $\frac{\partial c_j(\boldsymbol{x})}{\partial \phi_{lm}} \neq 0$ only for $j = m$, which is:

$$\frac{\partial c_m(\boldsymbol{x})}{\partial \phi_{lm}} = -\sigma\left(A_m(\boldsymbol{x}) - T_m\right) \frac{\partial b_m^+(\boldsymbol{x})}{\partial \phi_{lm}} = -\sigma\left(A_m(\boldsymbol{x}) - T_m\right)\left(1 - \mu_m(\boldsymbol{x})\right) x_l x_m$$

Using this to simplify Equation (14), we get:

$$-\frac{\partial \log \gamma_q(\boldsymbol{x})}{\partial \phi_{lm}} = (1 - \mu_m(\boldsymbol{x})) x_l x_m (1 - \sigma\left(A_m(\boldsymbol{x}) - T_m\right)) = (1 - \mu_m(\boldsymbol{x})) x_l x_m \sigma\left(T_m - A_m(\boldsymbol{x})\right) \tag{15}$$

The corresponding equation for $T_m$ is:

$$\frac{\partial \log \gamma_q(\boldsymbol{x})}{\partial T_m} = \sigma\left(A_m(\boldsymbol{x}) - T_m\right) \tag{16}$$

□

### 7.1 The Variational Equation and its Gradients

This subsection contains a review of the variational equation and its gradients. Let $\boldsymbol{h}, \boldsymbol{v}$ be hidden/visible parts of a random vector. Let $\theta$ be a scalar parameter for their joint distribution, $p(\boldsymbol{v}, \boldsymbol{h})$.

$$\begin{aligned}\frac{\partial \log p(\boldsymbol{v})}{\partial \theta} = \frac{1}{p(\boldsymbol{v})} \frac{\partial p(\boldsymbol{v})}{\partial \theta} &= \frac{1}{p(\boldsymbol{v})} \sum_{\boldsymbol{h}} \frac{\partial p(\boldsymbol{v}, \boldsymbol{h})}{\partial \theta} = \frac{1}{p(\boldsymbol{v})} \sum_{\boldsymbol{h}} p(\boldsymbol{h}, \boldsymbol{v}) \frac{\partial \log p(\boldsymbol{v}, \boldsymbol{h})}{\partial \theta} \\ &= \sum_{\boldsymbol{h}} p(\boldsymbol{h}|\boldsymbol{v}) \frac{\partial \log p(\boldsymbol{v}, \boldsymbol{h})}{\partial \theta} = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v})}\left[\frac{\partial \log p(\boldsymbol{h}, \boldsymbol{v})}{\partial \theta}\right]\end{aligned}$$

Suppose, we use a tractable $q(\boldsymbol{h}|\boldsymbol{v})$ to compute the above expectation instead:

$$\frac{\partial \log p(\boldsymbol{v})}{\partial \theta} = \mathbb{E}_{p(\boldsymbol{h}|\boldsymbol{v})}\left[\frac{\partial \log p(\boldsymbol{h}, \boldsymbol{v})}{\partial \theta}\right] = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\frac{\partial \log p(\boldsymbol{h}, \boldsymbol{v})}{\partial \theta}\right] + \frac{\partial \mathtt{KL}(q(\boldsymbol{h}|\boldsymbol{v})||p(\boldsymbol{h}|\boldsymbol{v}))}{\partial \theta} \tag{17}$$

Even if we solve the inference problem approximately accurately (i.e. with $q$ close to $p$), we may be off the mark when attempting to improve the model because of the second term. To obtain the second equality of Equation (17), consider:

$$\begin{aligned}\log p(\boldsymbol{v}) &= \log p(\boldsymbol{v}, \boldsymbol{h}) + \log p(\boldsymbol{h}|\boldsymbol{v}) && \text{for any } \boldsymbol{h} \\ &= \sum_{\boldsymbol{h}} q(\boldsymbol{h}|\boldsymbol{v}) \log p(\boldsymbol{h}, \boldsymbol{v}) - \sum_{\boldsymbol{h}} q(\boldsymbol{h}|\boldsymbol{v}) \log p(\boldsymbol{h}|\boldsymbol{v}) && \text{for any distribution } q \text{ on } \boldsymbol{h} \\ &= \sum_{\boldsymbol{h}} q(\boldsymbol{h}|\boldsymbol{v}) \log \frac{p(\boldsymbol{h}, \boldsymbol{v})}{q(\boldsymbol{h}|\boldsymbol{v})} + \sum_{\boldsymbol{h}} q(\boldsymbol{h}|\boldsymbol{v}) \log \frac{q(\boldsymbol{h}|\boldsymbol{v})}{p(\boldsymbol{h}|\boldsymbol{v})} \\ &= \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\log \frac{p(\boldsymbol{h}, \boldsymbol{v})}{q(\boldsymbol{h}|\boldsymbol{v})}\right] + \mathtt{KL}(q(\boldsymbol{h}|\boldsymbol{v})||p(\boldsymbol{h}|\boldsymbol{v}))\end{aligned}$$

Denote the first term above as a function of $p, q, \boldsymbol{v}$ where $\mathtt{H}(.)$ denotes the Shannon entropy:

$$\mathcal{L}(p, q, \boldsymbol{v}) \triangleq \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\log \frac{p(\boldsymbol{h}, \boldsymbol{v})}{q(\boldsymbol{h}|\boldsymbol{v})}\right] = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\log p(\boldsymbol{h}, \boldsymbol{v})\right] + \mathtt{H}\left(q(\boldsymbol{h}|\boldsymbol{v})\right) \tag{18}$$

Let $\phi$ be a parameter in the inferential/recognition density $q$. The variational approach to ML estimation involves gradient ascent on $\mathcal{L}$ for the entire collection of $\theta$ and $\phi$ simultaneously:

$$\frac{\partial \mathcal{L}(p, q, \boldsymbol{v})}{\partial \theta} = \mathbb{E}_{q(\boldsymbol{h}|\boldsymbol{v})}\left[\frac{\partial \log p(\boldsymbol{h}, \boldsymbol{v})}{\partial \theta}\right] \quad \text{and} \quad \frac{\partial \mathcal{L}(p, q, \boldsymbol{v})}{\partial \phi} = -\frac{\partial \mathtt{KL}_{\boldsymbol{v}}(q||p)}{\partial \phi} \tag{19}$$

To justify the gradient wrt $\phi$, note that $\log p(\boldsymbol{v}) = \mathcal{L}(p, q, \boldsymbol{v}) + \mathtt{KL}_{\boldsymbol{v}}(q||p)$ is independent of $q$. Therefore, we can interpret Equations (19) as compensation for dropping the term $\frac{\partial \mathtt{KL}_{\boldsymbol{v}}(q||p)}{\partial \theta}$ in Equation (17) for gradient ascent on $\log p(\boldsymbol{v})$ wrt $\theta$, by instead doing gradient ascent for an expanded variational collection that includes both $\theta$ and $\phi$ on $\mathcal{L}(p, q, \boldsymbol{v})$, by estimating the sensitivity of the KL distance with respect to the sampling density $q$ instead.

## 8 Acknowledgements

I would like to thank anonymous reviewers for feedback and Prof. Dale Schuurmans for his support.

## References

[1] Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

[2] Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1242–1250, 2014.

[3] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[4] Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

[5] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic back-propagation and variational inference in deep latent gaussian models. *arXiv preprint arXiv:1401.4082*, 2014.

[6] Ross D Shachter and Mark Alan Peot. Simulation approaches to general probabilistic inference on belief networks. *arXiv preprint arXiv:1304.1526*, 2013.

[7] Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In *Advances in neural information processing systems*, pages 3048–3056, 2013.

[8] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn.*, 1(1-2):1–305, January 2008.